Shortest paths in presence of node or link failures

Surender Baswana

Indian Institute of Technology Kanpur, India

Shortest Paths Problem

Problem Domain

Graph $G = (V, E)$, with $n = |V|$, $m = |E|$, $\omega : E \rightarrow R$.
$P(u, v)$: shortest path from u to v.
$\delta(u, v)$: distance from u to v.

Shortest Paths Problem

Single Source Shortest Paths (SSSP)

- *Positive edge weights:*
  Dijkstra's algorithm : $O(m + n \log n)$ time, $O(n)$ space.
- *Negative edge weights (but no negative cycle):*
  Bellman Ford algorithm : $O(mn)$ time, $O(n)$ space.

All-Pairs Shortest Paths (APSP)

- Floyd and Warshal Algorithm : $O(n^3)$
- Johnson's algorithm : $O(mn + n^2 \log n)$
- Pettie [2004] : $O(mn + n^2 \log \log n)$

Shortest paths in planar graphs

Planar graph

A graph is said to be planar if its vertices can be embedded on a sphere so that no two edges cross each others.

Research on SSSP for planar graphs

For possibly negative weights

- Late 70's : $O(n^{1.5})$
- ...
- Klein[2006] : $O(n \log n)$

Key ingredients

• Topology.
• Small size separator.

Shortest paths in presence of vertex failure

Algorithmic Objective

Construct a data-structure that supports following query.

- report $P(x, y, z)$: the shortest path from x to y in $G \backslash \{z\}$.
- report $\delta(x, y, z)$: the length of the path $P(x, y, z)$.

Motivation and applications

A model for dynamic shortest paths

1. At any time a subset $S \subset V$ of at most t vertices may be down.
2. The set S may keep changing but $|S| \leq t$ holds always.

Other applications

- k-shortest paths problem
- most vital node or link

Single source shortest paths in presence of vertex failure

Trivial upper bound

- preprocessing time: $O(mn)$
- space: $O(n^2)$

Lower bounds for directed graphs

- preprocessing time: $\Omega(m\sqrt{n})$
- space: $\Omega(n^2)$

Replacement paths problem for a source-destination pair

Replacement paths problem for a (r, t) pair

Problem definition

Given an undirected graph, a source r and destination t, compute $P(r, t, e)$ efficiently for all $e \in P(r, t)$.

Solution

$O(m)$ time and $O(n)$ space solution
Gupta and Malik [1989], Hershberger and Suri [2001]

Notations used

T_r : shortest path tree rooted at r.

$T_r(x)$: subtree of T_r rooted at x.

Replacement paths problem for a (r, t) pair

Key Ideas

1. Revisiting the shortest paths problem
2. Investigating the properties of $P(r, t, e)$
3. Deriving key observations
4. Using elementary data structure

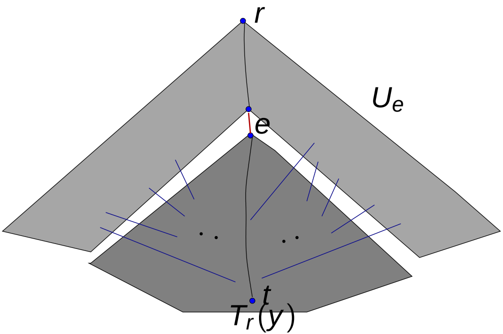
Handling an edge failure

Revisiting the shortest paths problem

Recall Dijkstra's algorithm ...

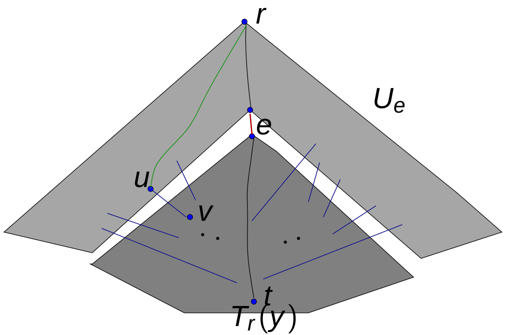
1. optimal subpath property
2. use of Heap data structure

Investigating properties of $P(r, t, e)$



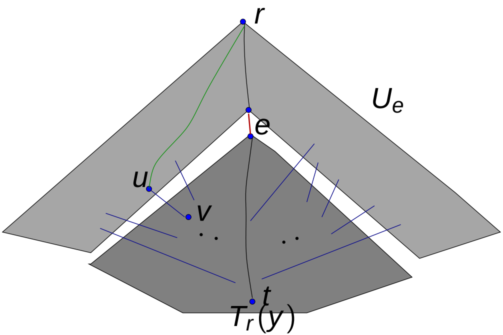
How does the path $P(r, t, e)$ look like ?

Observation 1



Once $P(r,t,e)$ leaves U_e, it never enters U_e again

Observation 2



$P(v,t,e)$ is the same as $P(v,t)$

Replacement paths problem for (r, t) pair

Key idea

For an edge $e = (x, y)$

$$\delta(r, t, e) = \min_{(u,v) \in E, u \in U_e, v \in T_r(y)} \delta(r, u) + \omega(u, v) + \delta(t, v)$$

Replacement paths problem for (r, t) pair

Key idea

For an edge $e = (x, y)$

$$\delta(r, t, e) = \min_{(u,v) \in E, u \in U_e, v \in T_r(y)} \delta(r, u) + \omega(u, v) + \delta(t, v)$$

solution lies in classical SSSP itself !!

Replacement paths problem for (r, t) pair

An $O(m)$ time and $O(n)$ space solution

- build shortest path tree T_r rooted at source r
- build shortest path tree T_t rooted at destination t
- use Heap on *crossing edges* with *suitable weights*.

All-pairs replacement paths problem

Problem Definition

Compute a compact data structure for reporting $P(r,t,x)$ and/or $\delta(r,t,x)$ for all $r,t,x \in V$ in optimal time.

Solution

- Demetrescu et al. [SICOMP 2008]
  $\tilde{O}(n^2)$ storage-space and $O(mn^2 \operatorname{polylog} n)$ processing time.

Problem Definition

Compute a compact data structure for reporting $P(r, t, x)$ and/or $\delta(r, t, x)$ for all $r, t, x \in V$ in optimal time.

Solution

- Demetrescu et al. [SICOMP 2008]
  $\tilde{O}(n^2)$ storage-space and $O(mn^2 \operatorname{polylog} n)$ processing time.
- Bernstein and Karger [STOC 2009]
  Improved processing time to $O(mn \operatorname{polylog} n)$.

Overcoming challenges through Collaboration

A toy problem :

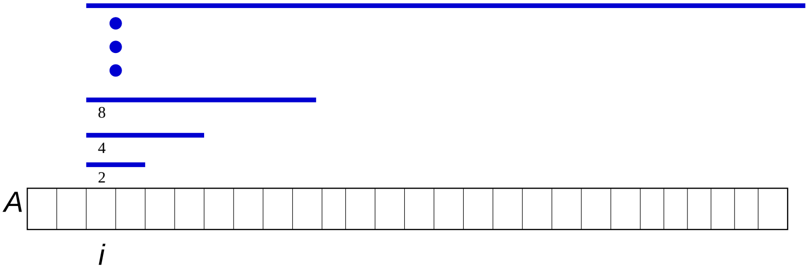
Given an array A storing n numbers, design a data structure to to answer query of the following kind

- report_min(A, i, j): smallest element from $\{A[i], A[i+1], ..., A[j]\}$.

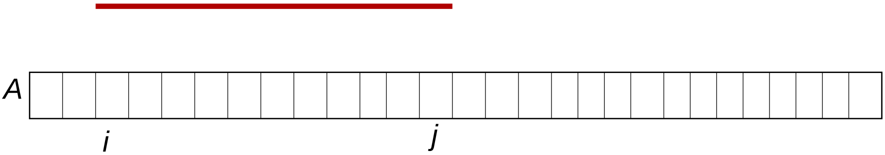
Trivial solution

Build an $n \times n$ table M where $M[i, j]$ stores the smallest element from $\{A[i], A[i+1], ..., A[j]\}$.

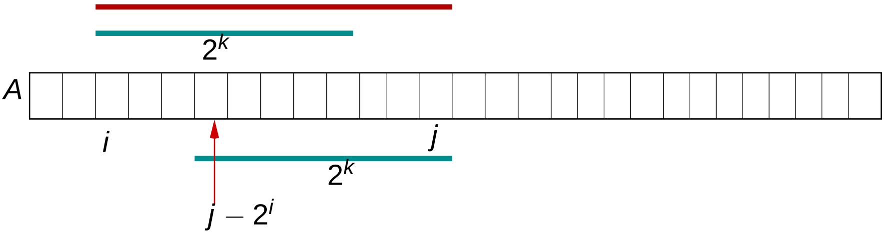
Solving the toy problem through collaboration



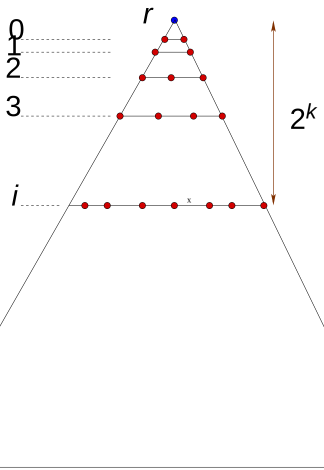
Solving the toy problem through collaboration



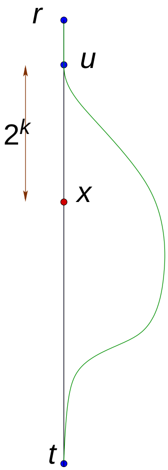
$O(n \log n)$ space and $O(1)$ query time solution



Solving all-pairs replacement paths problem



Solving all-pairs replacement paths problem



Solving all-pairs replacement paths problem

For each $u \in V$ and $i \in [0, \log_2 n]$, do

- Compute and store $\delta(u, v, x)$ for all x lying at level 2^i in G.
- Compute and store $\delta(u, v, x)$ for all x lying at level 2^i in G^r. (guess why ...)
- Compute and store $\delta(u, v, P)$ for all paths P starting from a vertex at level 2^{i-1} to level 2^i. (guess why ...)

Recent results on replacement paths

- Efficient solution for Approximate replacement paths
- Efficient solution for Planar graphs

Single source shortest paths in presence of vertex failure

Trivial upper bound

- preprocessing time: $O(mn)$
- space: $O(n^2)$

Lower bounds

- preprocessing time: $\Omega(m\sqrt{n})$
- space: $\Omega(n^2)$

New results for approximate replacement paths

New results for approximate replacement paths

Compact data structures for

- Single Source approximate shortest paths avoiding any failed vertex

New results for approximate replacement paths

Compact data structures for

- Single Source approximate shortest paths avoiding any failed vertex
- All-pairs approximate shortest paths avoiding any failed vertex

New results for approximate replacement paths

Compact data structures for

- Single Source approximate shortest paths avoiding any failed vertex
- All-pairs approximate shortest paths avoiding any failed vertex

Optimality !!

the size of our data structures nearly match their best static counterpart. [STACS 2010]

Single source version

best static result

keep a shortest path tree, space = $O(n)$

Single source version

best static result

keep a shortest path tree, space = $O(n)$

New result I

Single source approx. shortest paths avoiding a failed vertex

	Stretch	Space
Weighted graph	3	$O(n \log n)$

Single source version

best static result

keep a shortest path tree, space = $O(n)$

New result I

Single source approx. shortest paths avoiding a failed vertex

	Stretch	Space
Weighted graph	3	$O(n \log n)$
Unweighted graph	$(1+\epsilon)$	$O(n/\epsilon^3 \log n)$

All-pairs version

best static result by Thorup and Zwick [JACM 2005]

All-pairs $(2k-1)$-Approximate shortest paths oracle

Stretch	Space	Query time
$(2k-1)$	$O(kn^{1+1/k})$	$O(k)$

All-pairs version

best static result by Thorup and Zwick [JACM 2005]

All-pairs $(2k-1)$-Approximate shortest paths oracle

Stretch	Space	Query time
$(2k-1)$	$O(kn^{1+1/k})$	$O(k)$

New result II

For *unweighted* graphs, an oracle capable of handling single vertex failure

All-pairs version

best static result by Thorup and Zwick [JACM 2005]

All-pairs $(2k-1)$-Approximate shortest paths oracle

Stretch	Space	Query time
$(2k-1)$	$O(kn^{1+1/k})$	$O(k)$

New result II

For *unweighted* graphs, an oracle capable of handling single vertex failure

Stretch	Space	Query time
$(2k-1)(1+\epsilon)$	$O(\frac{k}{\epsilon^4}n^{1+1/k}\log n)$	$O(k)$

New results for planar graphs [unpublished]

single source

- Preprocessing time: $O(n \log^4)$
- Space: $O(n \log^4 n)$
- Query time: $O(\log^2 n)$

single source

- Preprocessing time: $O(n\log^4)$
- Space: $O(n\log^4 n)$
- Query time: $O(\log^2 n)$

All-pairs

- Preprocessing time: $O(n\sqrt{n})$
- Space: $O(n\sqrt{n})$
- Query time: $O(\sqrt{n})$

Open problems

- For weighted graphs, is it possible to have single source oracle with $O(n)$ space but stretch better than 3 ?

Open problems

- For weighted graphs, is it possible to have single source oracle with $O(n)$ space but stretch better than 3 ?
- Extending the results of planar graphs to bounded genus graphs.